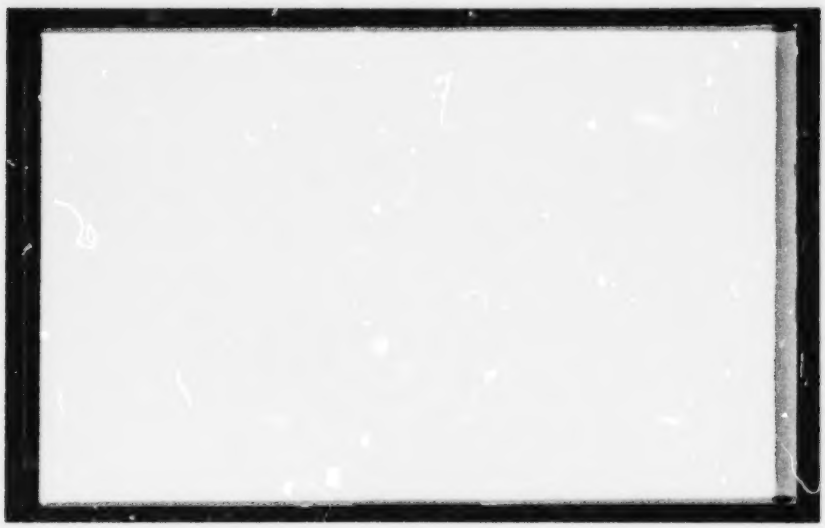
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

SUBTOUR ELIMINATION ALGORITHMS FOR THE SYMMETRICAL TRAVELLING SALESMAN PROBLEM AND ITS VARIANTS

by

Gilbert Laporte* and Yves Nobert**

* Ecole des Hautes Etudes Commerciales de Montréal

** Université de Sherbrooke

Université de Montréal

Centre de recherche sur les transports - Publication # 161

Ecole des Hautes Etudes Commerciales de Montréal -
Rapport de recherche no 80-04

Janvier 1980

Dépôt légal - 1er trimestre 1980

ABSTRACT

Several routing problems are shown to have the same structure. Methods for their solution can be built around algorithms for the symmetrical travelling salesmen problem. All problems are formulated as integer linear programs and constraint relaxation is used for their solution. This paper examines three basic cases: the m-travelling salesman problem (m-TSP), the truck dispatching problem and the problem of finding the best depot location for the m-TSP.

1. INTRODUCTION

In Operational Research, several routing and scheduling problems arising from different practical contexts can be shown to have similar structures. Methods for their solution can be built around algorithms for the more basic travelling salesman problem (TSP). This problem consists in finding the shortest Hamiltonian circuit passing through each of n points exactly once. The problem can be thought as that of a salesman wishing to find the shortest tour through cities 1,2,...,n where each city is only visited once. Let c_{ij} denote the distance from point i to point j: if $c_{ij} = c_{ji}$ for all pairs of points (i,j) the problem is symmetrical; otherwise it is asymmetrical. If $c_{ik} + c_{kj} \geq c_{ij}$ (for all i,j,k), the problem is Euclidean.

Recent papers by Miliotis [8,9] and Land [3] have shown that by using an appropriate form of constraint relaxation, the TSP can be solved efficiently by integer linear programming. Their algorithms seem particularly powerful for the solution of symmetrical TSP's. Other types of algorithms appear to be more efficient for asymmetrical problems [1,10]. Our aim is to show how some extensions of the symmetrical TSP can be formulated and solved efficiently as integer linear programs. These formulations all include the simple TSP as a special case. For the sake of brevity, we shall only consider symmetrical problems in this paper. All our formulations can easily be converted to handle asymmetrical problems. There is virtually no limit to the number of types of problems which can be formulated as TSP extensions and we shall concentrate our three basic cases: the m-TSP, the truck dispatching problem, the problem of finding the best depot location for the m-TSP.

Since the main object of this paper is to outline relationships between various formulations, computational results have been kept at a minimum. More detailed results are reported in [3,4,5,6,8,9].

2. THE SYMMETRICAL TRAVELLING SALESMAN PROBLEM

Along with several authors, Miliotis [8,9] and Land [3] use the following formulation for the symmetrical TSP:

(P1) $$\text{minimize} \sum_{i<j} c_{ij} x_{ij}$$

subject to

(D1) $$\sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad (k = 1,\ldots,n)$$

(S1) $$\sum_{i,j\in S} x_{ij} \leq |S| - 1 \qquad (3 \leq |S| \leq n-3;\ S \subsetneq \{1,\ldots,n\})$$

(I1) $$x_{ij} = 0 \text{ or } 1 \qquad (i < j;\ i,j = 1,\ldots,n)$$

In this problem, there are three types of constraints.

- The degree constraints (D1) merely specify that each node must be incident to two arcs.
- The subtour elimination constraints (S1) prevent the formation of Hamiltonian circuits involving less than n nodes. It is not necessary to specify such constraints for $|S| = 1,2,n-2$ or $n-1$ since such cases are automatically excluded by the presence of constraints (D1) and (I1).

When (D1) and (I1) are imposed, the following subtour elimination constraints are equivalent to (S1):

(S1') $$\sum_{\substack{i\in S,\ j\in\bar{S} \\ \text{or } j\in S,\ i\in\bar{S}}} x_{ij} \geq 2 \qquad (3 \leq |S| \leq n-3;\ S \subsetneq \{1,\ldots,n\})$$

This equivalence can be extended to a more general case:

$$(SG1) \qquad \sum_{i,j \in S} x_{ij} \leq |S| - r \qquad (3 \leq |S| \leq n-3;\ S \subseteq \{1,\ldots,n\})$$

is indeed equivalent to

$$(SG1') \qquad \sum_{\substack{i \in S,\ j \in \bar{S} \\ \text{or } j \in S,\ i \in \bar{S}}} x_{ij} \geq 2r \qquad (3 \leq |S| \leq n-3;\ S \subseteq \{1,\ldots,n\})$$

whenever $r \in \{1,\ldots, |S|-2\}$ and (D1), (I1) are imposed.

Although (SG1) and (SG1') are equivalent, (SG1) will often be preferred to (SG1') since these constraints tend to contain less coefficients and require therefore less computer memory.

- The last constraints (I1) are the integrity constraints; variables x_{ij} must be equal to 1 if and only if arc (i,j) is included in the optimal solution. These constraints can be satisfied either by using a branch and bound algorithm [8] or Gomory cuts [3,9].

Miliotis [8,9] has devised a number of efficient algorithms in which (P1) is first solved without the presence of (S1) and (I1). The relaxed constraints are only introduced into the problem as they are violated. Miliotis presents two relaxation strategies:

- the straight algorithm: (P1) is first solved with constraints (D1). (I1) constraints are gradually introduced into the problem until an integer solution is found. Checks on subtours are then made and one or many (S1) constraints may then be introduced into the problem. This may of course cause the loss of integrity which will have to be regained. In a cutting

planes algorithm, subtour elimination constraints and Gomory cuts are gradually added until a feasible solution has been obtained. In a branch and bound algorithm, the search for a solution ends when all branches of the tree have been explored.

- The reverse algorithm: as in the straight algorithm, constraints (S1) and (I1) are first relaxed. However, checks for the presence of illegal subtours are made before an integer solution has been obtained. In this context, an illegal subtour is defined in a loose sense, two nodes i and j being linked together whenever $x_{ij} > 0$.

Miliotis concludes that when cutting planes are used to reach integrity, the reverse algorithm produces optimal solutions to (P1) in less time than the straight algorithm. If constraints (I1) are to be satisfied by the introduction of cutting planes, very few Gomory cuts and subtour elimination constraints need be generated [9].

The reverse algorithm has later been improved by Land [3] who uses variable relaxation as well as constraint relaxation and stores constraints more efficiently. Computational results obtained for 10 Euclidean and 3 non-Euclidean 100-city problems show the efficiency of the relaxation methods. Computation times are much higher in the case of Euclidean problems and also have a larger variance (times varied from 34.6 to 919.7 seconds). The 3 non-Euclidean problems were solved in less than 25 seconds on the London University CDC 7600 computer (FTN compiler).

3. THE m-TRAVELLING SALESMAN PROBLEM (m-TSP)

In the m-TSP, m salesmen based at city 1 must visit the n-1 remaining cities in the most economical way. As in the TSP, each city must be visited only once; city 1 is of course visited m times. m can be fixed a priori or let free, i.e. its optimal value is to be found simultaneously with the best route. In the latter case, it is easy to verify that the optimal value for m will be 1 in all problems where

$$c_{ij} \leq c_{1i} + c_{1j} \qquad (i < j \; ; \; i,j = 2,\ldots,n)$$

Several authors (see for example [7]) have suggested transforming the m-TSP with n cities into a 1-TSP with $n+\bar{m}-1$ cities where $\bar{m}$ is the (fixed) number of salesman or an upper bound on the number of salesmen. They cannot handle the case where m is free, unless of course $\bar{m} = n-1$, in which case the size of the original problem is nearly doubled. As Laporte and Nobert [5] have shown, this transformation is not only restrictive but it substitutes to the original problem a more difficult problem, whereas one could take advantage of the fact that when m is free, the m-TSP is a relaxation of the 1-TSP. Formulating and solving the m-TSP directly appears to be more flexible and more efficient. The formulation is as follows:

(P2) $$\text{Minimize} \quad \sum_{i<j} c_{ij} x_{ij} + fm$$

subject to

(D2)
$$\begin{cases} \sum_{j=2}^{n} x_{ij} = 2m \\ \sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad (k = 2,\ldots,n) \end{cases}$$

$$(S2) \qquad \sum_{\substack{i<j \\ i,j\in S}} x_{ij} \leq |S| - 1 \qquad (3 \leq |S| \leq n-2;\ S \subset \{2,\ldots,n\})$$

$$(I2) \quad \begin{cases} x_{ij} = 0 \text{ or } 1 & (1 < i < j) \\ x_{1j} = 0, 1 \text{ or } 2 & (j = 2,\ldots,n) \\ m \geq 1 \text{ and integer} \end{cases}$$

Here f is a fixed cost on a salesman. m is a variable (if m is free) or a constant (if m is fixed). In the latter case, the last constraint and the last term of the objective function are superfluous.

(P2) is similar to (P1) except that

- the degree constraint for node 1 is different,
- in (S2), it is necessary to write $3 \leq |S| \leq n-2$ instead of $3 \leq |S| \leq n-3$ since otherwise the following situation would not be precluded: $n = 6$, $x_{1,2} = 2$, $x_{3,4} = x_{4,5} = x_{5,6} = x_{3,6} = 1$.

  However, since subtours including the depot are allowed, it is only necessary to specify $S \subseteq \{2,\ldots,n\}$ as long as $3 \leq |S| \leq n-2$. Note that in (P1), it would have been possible to restrict S to $\{2,\ldots,n\}$ since by excluding all subtours not including city 1, one is automatically excluding those including city 1.
- x_{1j} may be equal to 2, i.e. return trips from the depot to any one city arc allowed.

It is easy to see how (P2) reduces to (P1) when m is fixed at 1.

With some modifications, we have applied Miliotis' straight and reverse algorithms to (P2). Our results [5] confirm the superiority of the reverse algorithm. When m is free, the algorithm performs at least as well as in the case of the TSP. On the average, problems of up to 100 cities have been solved in just over 200 seconds on the University of Montreal CYBER 173 computer. When m is fixed, solution times decrease as m increases. In all cases, solving the m-TSP directly as we did was much more efficient than transforming it into a 1-TSP [7], enabling in some instances, time savings of up to 88% to be made.

4. THE TRUCK DISPATCHING PROBLEM (TDP)

Let us now assume that in the m-TSP, salesmen are replaced by trucks. Each is capable of carrying a weight at most equal to D. Goods weighing d_i must be delivered to city i. Since trucks are based at city 1, one can set $d_1 = 0$ without loss of generality. Several attempts to solve this problem have been reported in the Operational Research literature (see for example [2]), but so far little success has been achieved. We suggest the following formulation for the TDP.

(P3) $$\text{Minimize} \quad \sum_{i<j} c_{ij} x_{ij} + fm$$

subject to

(D3) $$\begin{cases} \sum_{j=2}^{n} x_{1j} = 2m \\ \sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \qquad (k = 2,\ldots,n) \end{cases}$$

(S3) $$\sum_{\substack{i<j \\ i,j\in S}} x_{ij} \leq |S| - \left\{ \frac{\sum_{i\in S} d_i}{D} \right\} \qquad (3 \leq |S| \leq n-2;\ S \subseteq \{2,\ldots,n\})$$

$$(I3)\quad \begin{cases} x_{ij} = 0 \text{ or } 1 & (1 < i < j) \\ x_{1j} = 0, 1 \text{ or } 2 & (j = 2,\ldots,n) \\ m \geq 1 \text{ and integer} \end{cases}$$

In this formulation, all variables and coefficients are defined in the same way as in (P2). Here, $\{x\}$ denotes the smallest integer greater than or equal to x.

Constraints (D3) and (I3) are the same as constraints (D2) and (I2). Since the m-TSP can be viewed as a TDP with

$$d_i = 1 \quad (i = 2,\ldots,n) \quad \text{and} \quad D = n,$$

it then follows that in the m-TSP,

$$\left\{\frac{\sum_{i\in S} d_i}{D}\right\} = 1 \quad \text{whenever} \quad S \subset \{2,\ldots,n\}$$

Therefore, constraints (S2) appear as a special case of constraints (S3).

It remains to show that constraints (S3) are valid constraints for the TDP: for any subset S of $\{2,\ldots,n\}$,

$$\left\{\frac{\sum_{i\in S} d_i}{D}\right\} \text{ is the minimum number of trucks needed to visit}$$

points in S. It therefore follows that

$$(S3')\quad \sum_{\substack{i\in S,\ j\in\bar{S} \\ \text{or } j\in S,\ i\in\bar{S}}} x_{ij} \geq 2 \cdot \left\{\frac{\sum_{i\in S} d_i}{D}\right\} \qquad (3 \leq |S| \leq n-2;\ S \subset \{2,\ldots,n\})$$

From the equivalence between (SG1) and (SG1'), constraints (S3') can then be substituted by constraints (S3).

One can solve the TDP by applying to (P3) a relaxation algorithm similar to those used for (P1) and (P2). However, the relaxation does not appear to be as efficient as previously and a far greater number of subtour elimination constraints have to be introduced. This particular behavior of the algorithm can be explained in the following way: as long as constraints (S3) are relaxed, the algorithm will produce solutions containing a small number of subtours involving many cities, thus minimizing the total distance travelled by the m trucks. The weight of any given subtour is thus likely to exceed D. Several subtour elimination constraints will therefore be required before a solution containing subtours which all include city 1 and whose weight does not exceed D is produced. Since many subtour elimination constraints will be required in most problems, the convergence of Gomory cuts to an integer solution tends to be slower. This has been confirmed by Laporte and Nobert [4]. We have so far applied the straight algorithm, using branch and bound to satisfy (I3). For problems of a given size, computation times vary immensely. Problems having up to 35 cities have been solved but there is no guarantee that our algorithm will solve any 35 city problems.

5. DISTANCE CONSTRAINTS

In some m-TSP's, it is required that the length of any subtour should not exceed a preset bound C. This problem (P4) can be formulated as (P3) except that constraints (S3) are replaced by

$$(S4) \qquad \sum_{\substack{i<j \\ i,j\in S}} x_{ij} \leq |S| - \left\lceil \frac{\ell(S \cup \{1\})}{C} \right\rceil \qquad (3 \leq |S| \leq n-2,\ S \subseteq \{2,\ldots,n\})$$

where $\ell(S)$ is a lower bound on the length of the shortest Hamiltonian circuit involving only points of S.

When C is sufficiently large (e.g. $C = \sum_{i<j} c_{ij}$),

$$\left\{\frac{\ell(S \cup \{1\})}{C}\right\} = 1 \quad \text{whenever} \quad S \subseteq \{2,\ldots,n\}.$$

and therefore (P2) is a special case of (P4).

To prove the validity of constraints (S4), we observe that any subset S of {2,...,n} must be linked to city 1 and will have to be visited by at least $\left\{\frac{\ell(S \cup \{1\})}{C}\right\}$ trucks. Therefore the following constraints can be imposed:

$$(S4') \qquad \sum_{\substack{i\in S,\ j\in\bar{S} \\ \text{or } j\in S,\ i\in\bar{S}}} x_{ij} \geq 2 \cdot \left\{\frac{\ell(S \cup \{1\})}{C}\right\} \quad \begin{array}{l}(3 \leq |S| \leq n-2; \\ S \subseteq \{2,\ldots,n\})\end{array}$$

As noted in section 2, these constraints are equivalent to (S4).

It is worth mentioning that if a branch and bound algorithm is used to satisfy (I3) in (P4), constraints (S4) are only valid for the subproblem to which they are applied and must be removed when backtracking. This is so because the value of $\ell(S \cup \{1\})$ is affected by the various variables which are fixed. In contrast, constraints (S3) remained valid throughout the solution of (P3).

The value of $\left\{\frac{\ell(S \cup \{1\})}{C}\right\}$ is easy to compute. Let us consider an optimal solution to the relaxed problem (P4) which contains at least one subtour. In this solution, (D3) and (I3) are satisfied. Let S be a subtour in this solution. If $1 \in S$, then

$$\ell(S \cup \{1\}) = \ell(S) = \sum_{i,j\in S} c_{ij}\, x_{ij}$$

(since the current solution is optimal). If $1 \notin S$, the length of the shortest Hamiltonian circuit involving only points of $S \cup \{1\}$ is not known but it is bounded below by $\ell(S)$ (for otherwise the current solution would not be optimal). Therefore, $\ell(S \cup \{1\})$ can be taken as $\sum_{i,j\in S} c_{ij}\, x_{ij}$.

6. SUBTOURS WITH A LIMITED NUMBER OF CITIES

In the m-TSP, one may specify the additional constraint that no subtour should contain more than K cities (apart from city 1). This problem (P5) is a special case of (P3) with D = K and $d_2 = d_3 = \ldots = d_n = 1$. Constraints (S3) then become:

$$(S5) \qquad \sum_{\substack{i<j \\ i,j \in S}} x_{ij} \leq |S| - \left\{\frac{|S|}{K}\right\} = [(1 - \tfrac{1}{K})\,|S|] \qquad (3 \leq |S| \leq n-2,\ S \subseteq \{2,\ldots,n\}).$$

where [x] denotes the largest integer less than or equal to x.

7. LOCATING THE DEPOT OPTIMALLY

In all of the problems just described, it is assumed that the depot location is known (city 1). We now consider the case where one wishes to locate the depot that will produce the shortest routes for m salesmen (here, m is fixed). One can of course consider an operating cost for each possible depot site. This problem is described in [6]. Let y_k be a binary variable which takes the value 1 if node k is used as a depot and the value 0 otherwise. Let f_k be the cost of operating a depot at node k. The integer linear programming formulation is as follows.

$$(P6) \qquad \text{Minimize} \quad \sum_{i<j} c_{ij} x_{ij} + \sum_{k=1}^{n} f_k y_k$$

subject to

$$(K6) \qquad \sum_{k=1}^{n} y_k = 1$$

$$(D6) \qquad \sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 + 2(m-1)y_k \qquad (k = 1,\ldots,n)$$

$$(S6) \qquad \sum_{\substack{i<j \\ i,j\in S}} x_{ij} \le |S| - 1 + (m-1) \sum_{k\in S} y_k \qquad (2 \le |S| \le n-2,\ S \subseteq \{1,\ldots,n\}).$$

$$(I6) \qquad x_{ij} = 0,\ 1 \text{ or } 2 \qquad (i < j)$$

$$y_k = 0 \text{ or } 1 \qquad (k = 1,\ldots,n)$$

Here, constraint (K6) indicates that there should be only one depot. Constraints (S6) eliminate subtours not including the depot. If S does not contain the depot or if m = 1, constraints (S6) are equivalent to constraints (S2.) If S contains the depot,

$$(|S| - 1 + m \sum_{k\in S} y_k)$$

is an upper bound on the number of arcs contained in S and used by a salesman. The computation of this bound assumes that all m salesmen tours only involve points in S. However, as long as $|S| \le n-2$, this upper bound cannot be reached as there would be no salesman left for points outside S. Hence

$$(|S| - 1 + (m-1) \sum_{k\in S} y_k)$$

is a valid upper bound.

It is worth noticing that (P6) reduces to (P2) when the depot is fixed.

Laporte and Nobert [6] have solved (P6) by using the straight algorithm and branch and bound to reach integrity. Problems having up to 50 cities have been solved in 600-700 seconds in the CYBER 173 computer. These times are of course much larger than those obtained for (P2) but solving (P6) is more efficient than solving n times an m-TSP, using all the nodes in turn as a depot.

REFERENCES

1. Carpaneto, G., Toth, P., "Some New Branching and Bounding Criteria for the Asymmetric Travelling Salesmen Problem", paper presented at the EURO III Conference, Amsterdam, April 1979.

2. Christofides, N., Mingozzi, A., Toth, P., "The Vehicle Routing Problem", Combinatorial Optimization, Wiley, 1979.

3. Land, A.H., "The Solution of Some 100-city Travelling Salesman Problems", paper presented at the Tenth International Symposium on Mathematical Programming, Montreal, August 1979.

4. Laporte, G., Nobert, Y., "The Truck Dispatching Problem", paper presented at the EURO III Conference, Amsterdam, April 1979.

5. Laporte, G., Nobert, Y., "A Cutting Planes Algorithm for the m-Salesmen Problem", Rapport de recherche 79-08, Ecole des Hautes Etudes Commerciales de Montréal, 1979.

6. Laporte, G., Nobert, Y., "An Exact Algorithm for Minimizing Routing and Operating Costs in Depot Location", to appear in European Journal of Operational Research.

7. Lenstra, J.K., Rinnooy Kan, A.H.G., "Some Simple Applications of the Travelling Salesman Problem", Operational Research Quarterly, Vol. 26, pp. 717-733, 1975.

8. Miliotis, P., "Integer Programming Approaches to the Travelling Salesman Problem", Mathematical Programming, Vol. 10, pp. 367-378, 1976.

9. Miliotis, P., "Using Cutting Planes to Solve the Symmetric Travelling Salesman Problem", Mathematical Programming, Vol. 15, pp. 177-188, 1978.

10. Smith, T.H.C., Srinivasan, V., Thompson, G.L., "Computational Performance of Three Subtour Elimination Methods for Solving Asymmetric Traveling Salesman Problems", Studies in Integer Programming, Annals of Mathematics 1, North-Holland, 1977.